

UNOFFICIAL COMMUNICATION FOR EXAMINER REVIEW ONLY – PLEASE DO NOT ENTER

REMARKS

**Rejection of Claims 1, 3, 5-11, 14-22, 25 and 27, 28-32 Under 35 U.S.C. §102(e)**

Claims 1, 3, 5-11, 14-22, 25 and 27, 28-32 stand rejected under 35 U.S.C. §102(e) as being anticipated by Scharefer (US Pub 2003/0084429. Withdrawal of this rejection is requested since Scharefer fails to teach or suggest all aspects of subject claims.

A single prior art reference anticipates a patent claim only if it *expressly or inherently describes each and every limitation* set forth in the patent claim. *Trintec Industries, Inc. v. Top-U.S.A. Corp.*, 295 F.3d 1292, 63 USPQ2d 1597 (Fed. Cir. 2002); *See Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). *The identical invention must be shown in as complete detail as is contained in the ... claim.* *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989).

Applicants' claimed invention relates to system and methodology to facilitate navigation for inexperienced and experienced programmers to create user interface automation. To this end, independent claim 1 recites *a navigation component that facilitates simulated user interface associated with an automation component based, at least in part, upon information stored in a map information store and information stored in a command information store.* Scharefer neither teaches nor suggests such novel aspects.

Scharefer provides systems and methods for table driven automation testing for performing functional testing of a software program. The system includes a GUI translator component to translate one or more GUI maps into a set of database tables, a data input component to facilitate entry and editing of test case data in the tables, and a test engine component for executing the software program based on a test case stored in the tables.

At page 4 of the Final Office Action, the Examiner incorrectly contends that Scharefer teaches *a navigation component that facilitates simulated user interface associated with an automation component.* The cited portion of reference provides for a graphical user interface translator component, a data input component, a test engine

## UNOFFICIAL COMMUNICATION FOR EXAMINER REVIEW ONLY – PLEASE DO NOT ENTER

component, a software controller component and an operating system. The GUI translator component translates one or more GUI maps into a set of database tables and a data input component facilitates creation of one or more test cases, which are input as data into the tables by the data input component (Paragraph 37, 41). The test engine component queries the tables to retrieve data for a test case and use one or more GUI maps along with the data for test case to call a function in the software controller. The software controller component receives instructions and data from the test engine component and transmits instructions and data to the software program thereby controlling the execution of software program (Paragraph 38). Test engine component uses the contents of GUI map stored in memory to determine which software controller component function to call to process the objects on the window that are described by the GUI map and also monitors results, received from the software controller about the execution of the software program (Paragraph 55). Hence Scharefer provides for table driven automation testing of a software program by translating one or more GUI maps into a set of database tables by a GUI translator component and then querying the tables to retrieve data for a test, and nowhere teaches or suggests *a navigation component that facilitates simulated user interface associated with an automation component*. The Examiner erroneously asserts that test engine component is same as navigation component. The test engine component of Scharefer provides for querying the tables to retrieve data for a test case and use one or more GUI maps along with the data for test case to call a function in the software controller and providing instructions and data to the software controller component. However the navigation component of the present invention *facilitates simulated user interface* and helps inexperienced programmers to create window type automation. *Navigation component and simulated user interface* facilitate the user to create UI automation without any substantial knowledge of functionality associated with the system and mitigates creation time.

At page 9 of the Office Action, the Examiner again erroneously asserts that Scharefer teaches storing data, commands and executables associated with the user interface separately which provides a modular system which can be modified without recompilation of executables, with respect to dependent claims 31 and 32. The cited portion of reference provides for storing GUI translator component, data input

## UNOFFICIAL COMMUNICATION FOR EXAMINER REVIEW ONLY – PLEASE DO NOT ENTER

component, test engine component, software controller component, and utilities on a secondary storage and loading into memory (Paragraph 43). Secondary storage is configured using any computer readable medium (Paragraph 45). If a user interface changes for a software program then the associated GUI map have to change as well as one or more of the tables in the database. The GUI map is updated to reflect the changes to the user interface by manually editing the GUI map file or by editing the GUI map by using GUI map editor. GUI translator component is used to modify the data in one or more of the tables to incorporate the user interface changes and can also create or delete tables (Paragraph 93). Hence Scharefer provides for storing data, commands and executables on same secondary storage. And modification of the user interface requires modification of the GUI map and tables in the database. Modification of GUI map and tables requires modification of the GUI map editor and GUI translator component, respectively, which are executables. However the present invention mitigates recompilation of executables as information (data and commands) associated with program flow are generally stored in simple text files, separate from executables, thus reducing the need to recompile the executables. Thus, modification of these text files only can produce new behavior for the executables. Update to the program flow only requires a modification to these text files and not the engine.

In view of at least the foregoing, it is clear that Scharefer fails to teach each and every aspect recited in the subject claims. Therefore, it is respectfully requested that this rejection be withdrawn.